Multi-Class Fuzzily Weighted Adaptive Boosting-based Self-Organising Fuzzy Inference Ensemble Systems for Classification

Xiaowei Gu and Plamen P Angelov, *Fellow, IEEE*

***Abstract*—Adaptive boosting (AdaBoost) is a widely used technique to construct a stronger ensemble classifier by combining a set of weaker ones. Zero-order fuzzy inference systems (FISs) are very powerful prototype-based predictive models for classification, offering both great prediction precision and high user-interpretability. However, the use of zero-order FISs as base classifiers in AdaBoost has not been explored yet. To bridge the gap, in this paper, a novel multi-class fuzzily weighted AdaBoost (FWAdaBoost)-based ensemble system with self-organising fuzzy inference system (SOFIS) as the ensemble component is proposed. To better incorporate SOFIS, FWAdaBoost utilises the confidence scores produced by SOFIS in both sample weight updating and ensemble output generation, resulting in more accurate classification boundaries and greater prediction precision. Numerical examples on a wide range of benchmark classification problems demonstrate the efficacy of the proposed approach.**

***Index Terms*—AdaBoost, ensemble classifier, fuzzy inference system, multi-class classification.**

I. Introduction

ENSEMBLE learning [1]–[3] is a powerful scheme aiming to construct a stronger classifier by merging many individual weaker classifiers. As an important topic in machine learning, it has been widely researched and successfully applied in many real-world applications such as image segmentation [4], face recognition [5], pattern analysis [6], data stream mining [7], etc.

Currently, bagging [8], [9] and boosting [10], [11] are the two mainstream methods for constructing ensemble classifiers [12]. Bagging is a parallel ensemble learning method, short for bootstrap aggregation. It firstly samples a standard training set with replacement into multiple subsets, each of which is used for training one base classifier. The obtained base classifiers are then combined by majority voting. In contrast, boosting trains a series of base classifiers consecutively with various distributed training data such that each base classifier complements its predecessors. The learned base classifiers are then combined by weighted majority voting with more weights given to the ones that perform better on training set. Therefore, boosting gives more focus to harder samples and can effectively reduce classification bias, leading to stronger generalisation capability and greater classification precision [12]–[14].

Adaptive boosting (AdaBoost) is one of the most widely used boosting methods [10], [12], [15] at present. The main advantage of AdaBoost over alternative boosting methods is its ability to maximise the classification margins, providing good generalisation [16]. To further improve the performance and robustness of AdaBoost, there have been many famous variants introduced. For example, a generalised version of AdaBoost called Real AdaBoost (ReAdaBoost) is presented in [15], [17] by involving prediction confidences in weight updating. A variant of ReAdaBoost called Parameterised AdaBoost (PAdaBoost) is proposed in [18]. This variant employs a weight updating strategy designed to penalise the misclassification of already correctly classified samples, which effectively increases the classification margins. Another version of ReAdaBoost named Gentle AdaBoost (GAdaBoost) is presented in [17] by leveraging adaptive Newton steps to minimise the training loss. Empirical studies show that GAdaBoost outperforms ReAdaBoost in noisy environments. To reduce overfitting, Modest AdaBoost (MAdaBoost) [19] is designed to focus more on training samples with lowest classification margin by decreasing the contributions of base classifiers that perform well on easy training samples only. As a result, MAdaBoost demonstrates better performance than ReAdaBoost and GAdaBoost. AveBoost2 [20] effectively prevents the weights of noisy samples from getting excessively large and mitigates the overfitting problem by averaging the training sample weights of current boosting iteration and previous iterations. FloatBoost is introduced in [21] to achieve the minimum classification errors through utilization of a backtrack mechanism. The backtrack mechanism removes these base classifiers that do not improve the overall classification precision each time when a new base classifier is added to the ensemble. Noise-detection AdaBoost (NDAdaBoost) improves the robustness of AdaBoost to noisy data by utilizing a noise-detection based loss function to adjust the weight distribution at each iteration [22]. In this way, the base classifiers are more focused on these misclassified noisy samples and correctly classified non-noisy samples. Robust AdaBoost (RoAdaBoost) [23] is developed using the majorisation-minimisation principle. Based on the truncated loss functions, this algorithm can reduce the impact of outliers and construct more sparse predictive models with improved prediction accuracy and variable selection. In addition, many variants of AdaBoost have been introduced for solving multi-class classification

X. Gu is with the School of Computing, University of Kent, Canterbury, CT2 7NZ, UK. email: X.Gu@kent.ac.uk.

P. Angelov is with the School of Computing and Communications, Lancaster University, Lancaster, LA1 4WA, UK. email: p.angelov@lancaster.ac.uk.

Corresponding author: Xiaowei Gu

Manuscript received XXXX XX, 2021; revised XXXX XX, 2021.

problems. The most popular multi-class AdaBoost variants include AdaBoost.M1 [24], AdaBoost.M2 [24], AdaBoost.MH [25], AdaBoost.MR [15], stagewise additive modelling using a multi-class exponential loss function (SAMME) [26] and Real SAMME (ReSAMME) [26].

The vast majority of existing studies employed mainstream classifiers, such as decision trees (DTs) [14], [15], artificial neural networks (ANNs) [27]–[29], support vector machines (SVMs) [12], [30] and k-nearest neighbours (KNNs) [22], [32] as the ensemble components for AdaBoost and its variants. The constructed ensemble models have demonstrated greater predictive abilities than single-model classifiers on a wide variety of benchmark problems [33]. However, the lack of transparency and explainability is a critical issue remaining unsolved for mainstream classifiers. It is well known that ANNs and SVMs are the typical types of "black box" model. DTs are usually considered as easy to interpret, but large ones with many branches are not easily interpretable. Without constraining the maximum depth, DT models learned from high-dimensional problems often are highly complex and pose presentation difficulties [34], [35]. KNNs use all the training samples to classify unlabelled data by following the "nearest neighbours" principle. Despite that the operating mechanism of KNNs is simple and straightforward, their interpretability is limited when applied to large-scale problems. With the rapid development and wide deployment of artificial intelligence techniques, the transparency and explainability of machine learning models have become increasingly important, especially for life-critical applications. Therefore, there is a high demand for developing novel high-precision ensemble models that are constructed from transparent, explainable classifiers.

Fuzzy (neuro-fuzzy) systems are powerful tools widely used for classification, offering both great precision and high interpretability. To date, there have been a few ensemble models proposed in the literature that employ fuzzy systems as ensemble components. For example, an AdaBoost-based ensemble classifier based on relational neuro-fuzzy systems is proposed in [36]. To improve the interpretability of the constructed ensemble models, the paper [36] further presents a method to combining the fuzzy rules learned by individual base models into a large rule base via normalisation. Later, this work [36] is extended in [37] by using logical-type neuro-fuzzy systems as ensemble components. To tackle high dimensional problems effectively, an ensemble framework combining multiple simpler fuzzy systems trained on different projections of data is proposed in [38]. An ensemble system based on eClass0 fuzzy classifiers [39] is introduced for streaming data classification [40]. pENsemble proposed in [41] is equipped with an evolving ensemble framework that can automatically initialise new base classifiers and prune stale ones to follow the changing patterns in data streams. A deep rule-based ensemble classifier combining deep convolutional neural networks and conventional fuzzy systems is proposed in [42] for remote sensing scene classification. With the aim of reducing the requirements for computational resources and improving efficiency of ensemble fuzzy classifiers, the influence of reducing the reference sets in collective decision-making via instance selection is investigated in [43]. In [44], a ensemble fuzzy classifier named SENFIS is proposed for big data classification. SENFIS is formed by a committee of fuzzy models learned from subsets of training data. In addition, to improve the overall classification accuracy, only the selected fuzzy models satisfying the prediction precision and diversity criteria can join the decision-making committee. An ensemble classification model combines a fuzzy classifier and a less interpretable but more accuracy classifier, i.e., ANN, is proposed in [45]. This ensemble aims to achieve high classification precision while maintaining its interpretability by utilising the so-called confidence-based voting strategy such that the fuzzy classifier serves as the main component and the second classifier will be activated only when the confidence level of the fuzzy classifier is low. In [46], the functional equivalence of Takagi–Sugeno–Kang (TSK) fuzzy systems to different regression approaches including stacking ensemble regression is studied, and it is shown that each IF-THEN rule in the fuzzy system is equivalent to a base model in stacking ensemble regression. Self-organising fuzzy inference system (SOFIS) [47], [48] is a recently introduced zero-order fuzzy inference system (FIS) for classification. It is capable of self-organising a set of prototype-based IF-THEN fuzzy rules from labelled training samples in a computationally efficient manner and classifying unlabelled samples with great precision. Utilising nonparametric statistic operators [49], SOFIS is free from prior assumptions on data generation models with predefined parameters, and operates on human-understandable prototypes identified from empirically observed data based on their ensemble properties and mutual distances. By forming Voronoi tessellations [50] around these prototypes with nearby data samples, SOFIS naturally self-calibrates highly precise decision boundaries in the data space for classification. Thanks to the prototype-based nature, its internal reasoning and decision-making processes are fully interpretable, explainable and traceable to/by human. Since being firstly introduced, SOFIS has been applied to many real-world problems such as fault diagnosis [51], industrial control [52]. Although preliminary works [53] on constructing ensemble classifiers with SOFIS via random subsampling (a variant of bagging) have reported encouraging results, the potential of SOFIS in ensemble models constructed by mainstream boosting methods has not been investigated, yet.

Therefore, in this paper, a novel AdaBoost-based ensemble system with SOFIS as its ensemble component is proposed for multi-class classification. The proposed ensemble classifier employs the multi-class AdaBoost algorithm SAMME as its implementation basis. However, to take advantages of the unique features of zero-order FISs [39], [47], [48], [54], SAMME is modified to utilise the confidence scores produced by the IF-THEN fuzzy rules of SOFIS in both sample weight updating and ensemble output generating, resulting in the new multi-class AdaBoost algorithm named fuzzily weighted AdaBoost (FWAdaBoost). Accordingly, the proposed new ensemble system is named as FWAdaBoost-based self-organising fuzzy inference ensemble system (FWAdaBoost-SOFIES). Compared with its predecessor (SAMME), FWAdaBoost gives extra weights to these challenging and easy-to-misclassify samples such that more precise classification

boundaries can be constructed. FWAdaBoost further integrates the levels of confidence that each ensemble component has towards its individual predictions into the final ensemble outputs, thereby effectively boosting the classification performance of FWAdaBoost-SOFIES. Numerical examples based on benchmark datasets demonstrated that FWAdaBoost can boost the performance of SOFIS to a greater extent than alternative boosting algorithms, and the constructed ensemble system is able to achieve the state-of-the-art classification accuracy outperforming the comparative algorithms involved in the experimental investigations.

To summarise, key features of this paper include:

1) a novel multi-class AdaBoost algorithm with both the sample weight updating and ensemble output generation schemes designed specifically for zero-order FISs by incorporating the confidence scores produced by base learners in system identification and decision-making;
2) a novel multi-class AdaBoost-based fuzzy inference ensemble system with highly transparent system structure and explainable decision-making process thanks to its prototype-based nature as well as greater prediction precision.

The remainder of this paper is organised as follows. Section II provides the theoretical background of this study. Technical details of the proposed approach are presented in Section III and a theoretical analysis on the bound of training error is presented in Section IV. Numerical examples serving as the proof of concept are given in Section V. This paper is concluded by Section VI.

II. Preliminaries

First of all, let $\{\boldsymbol{x}\}_K = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_K\}$ be a dataset in the real data space, $\Re^M$ with $\{y\}_K = \{y_1, y_2, \ldots, y_K\}$ being the corresponding class labels; $\boldsymbol{x}_k = [\boldsymbol{x}_{k,1}, \boldsymbol{x}_{k,2}, \ldots, \boldsymbol{x}_{k,M}]^T \in \Re^M$; M denotes the dimensionality; y_k is the class label of $\boldsymbol{x}_k$. It is assumed that $\{\boldsymbol{x}\}_K$ is composed of data samples of C different classes, namely, $y_k \in \{1, 2, \ldots, C\}$ for $\forall y_k \in \{y\}_K$. Based on the class labels, $\{\boldsymbol{x}\}_K$ can be split to C subsets, denoted as $\{\boldsymbol{x}\}^i_{K^i}$ $(i = 1, 2, \ldots, C)$. $\boldsymbol{x}^i_{K^i} = \{\boldsymbol{x}^i_1, \boldsymbol{x}^i_2, \ldots, \boldsymbol{x}^i_{K^i}\}$ is the set of data samples within $\{\boldsymbol{x}\}_K$ that belong to the ith class; K^i is the number of such samples. There are $\{\boldsymbol{x}\}^1_{K^1} \cup \{\boldsymbol{x}\}^2_{K^2} \cup \ldots \cup \{\boldsymbol{x}\}^C_{K^C} = \{\boldsymbol{x}\}_K$ and $\{\boldsymbol{x}\}^i_{K^i} \cap \{\boldsymbol{x}\}^j_{K^j} = \emptyset$ for $\forall i \neq j$. In addition, it is often observed that different samples of the same classes may share exactly the same values, i.e., $\boldsymbol{x}^i_n = \boldsymbol{x}^i_m$ and $n \neq m$. Therefore, without loss of generality, the set of unique samples of the ith class is denoted as $\{\boldsymbol{u}\}^i_{L^i} = \{\boldsymbol{u}^i_1, \boldsymbol{u}^i_2, \ldots, \boldsymbol{u}^i_{L^i}\} (\{\boldsymbol{u}\}^i_{L^i} \subseteq \{\boldsymbol{x}\}^i_{K^i})$ and the corresponding occurrence frequencies are denoted as $\{f\}^i_{L^i} = \{f^i_1, f^i_2, \ldots, f^i_{L^i}\}$; f^i_k is the occurrence frequency of $\boldsymbol{x}^i_k$; L^i is the number of unique data samples $(i = 1, 2, \ldots, C)$; and $\sum_{l=1}^{L^i} f^i_l = K^i$.

A. SOFIS

The general architecture of SOFIS is given by Fig. 1. It can be observed from this figure that SOFIS is composed of C zero-order AnYa type IF-THEN fuzzy rules (one rule per class) [55] formed by prototypes, which are given in the form of Eqn. (1) [47], [48].

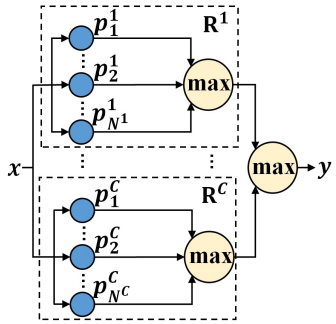

Fig. 1: Architecture of SOFIS [47], [48].

$$\mathbf{R}^i: \ IF\ (\boldsymbol{x} \sim \boldsymbol{p}^i_1)\ OR\ (\boldsymbol{x} \sim \boldsymbol{p}^i_2)\ OR\ ...\ OR\ (\boldsymbol{x} \sim \boldsymbol{p}^i_{N^i})\ THEN\ (class\ i) \tag{1}$$

where $\boldsymbol{x}$ is the input sample; "$\sim$" denotes similarity; $\boldsymbol{p}^i_j$ stands for the jth prototype of the ith class; and N^i is the number of identified prototypes of the ith class. These prototypes are highly representative samples in the data space and they help the classifier preserve the structure and underlying patterns of the original data.

One can see from Eqn. (1) that the premise part of AnYa type fuzzy rules is simplified to a more compact, objective and non-parametric form of prototypes. The prototypes of $\mathbf{R}^i$ are connected by the logic "OR" connectives, hence, $\mathbf{R}^i$ can be viewed as a parallel ensemble of multiple simpler fuzzy rules as follows $(j = 1, 2, ..., N^i)$:

$$\mathbf{R}^i_j: \ IF\ (\boldsymbol{x} \sim \boldsymbol{p}^i_j)\ \ THEN\ (class\ i) \tag{2}$$

Remark 1: By using these prototypes to attract nearby data samples and build Voronoi tessellations [50], SOFIS partitions the data space into non-overlapping, shape-free clusters. Each cluster belongs to a particular class, and the shared borders between clusters of different classes naturally form the decision boundaries for classification [47].

The identification and validation processes of SOFIS are summarised as follows. By default, it employs cosine dissimilarity as the distance measure, which is formulated as Eqn. (3) [48].

$$d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{1 - \cos(\theta_{ij})} = \frac{1}{\sqrt{2}} \parallel \frac{\boldsymbol{x}_i}{\parallel \boldsymbol{x}_i \parallel} - \frac{\boldsymbol{x}_j}{\parallel \boldsymbol{x}_j \parallel} \parallel \tag{3}$$

where θ_{ij} is the angle between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$; $\parallel \boldsymbol{x} \parallel$ is the Euclidean norm of $\boldsymbol{x}$. Therefore, to facilitate computation, all data samples are normalised by their corresponding Euclidean norm, namely, $\boldsymbol{x} \leftarrow \frac{\boldsymbol{x}}{\|\boldsymbol{x}\|}$ for $\forall x \in \{\boldsymbol{x}\}_K$ so that the cosine dissimilarity is simplified to Euclidean distance.

A. Identification process

The system identification process consists of the following three stages [47], [48].

Stage 1. Forming Voronoi tessellation from data

In this stage, multimodal density values at the unique data samples are firstly calculated using Eqn. (4) [49].

$$D^{MM}(\boldsymbol{u}^i_k) = \frac{f^i_k}{1 + \frac{\|\boldsymbol{u}^i_k - \bar{\boldsymbol{x}}^i\|^2}{1 - \|\bar{\boldsymbol{x}}^i\|^2}} \tag{4}$$

where $\bar{\boldsymbol{x}}^i$ is the arithmetic mean of $\{\boldsymbol{x}\}^i_{K^i}$, namely, $\bar{\boldsymbol{x}}^i = \frac{1}{K^i}\sum_{k=1}^{K^i}\boldsymbol{x}^i_k$.

Then, unique data samples of each class are ranked one-by-one in terms of their multimodal density values and mutual distances using Eqn. (5), re-denoted as $\{\boldsymbol{r}\}^i_{L^i} = \{\boldsymbol{r}^i_1, \boldsymbol{r}^i_2, \ldots, \boldsymbol{r}^i_{L^i}\}$ $(i = 1, 2, ..., C)$ [47], [48].

$$\begin{cases} \boldsymbol{r}^i_k = \underset{\boldsymbol{u}\in\{\boldsymbol{u}\}^i_{L^i}}{\arg\max}(D^{MM}(\boldsymbol{u})), & if\ k = 1 \\ \boldsymbol{r}^i_k = \underset{\substack{\boldsymbol{u}\in\{\boldsymbol{u}\}^i_{L^i};\\ \boldsymbol{u}\neq\boldsymbol{r}^i_1,\boldsymbol{r}^i_2,\ldots,\boldsymbol{r}^i_{k-1}}}{\arg\min} (\parallel \boldsymbol{u} - \boldsymbol{r}^i_{k-1} \parallel^2), & else \end{cases} \tag{5}$$

After the ranking operation, local maxima of multimodal density, denoted as $\{\boldsymbol{l}\}^i_{Q^i}$ ($\{\boldsymbol{l}\}^i_{Q^i} \leftarrow \{\boldsymbol{r}^i_1\}$, $i = 1, 2, \ldots, C$), are identified from unique data samples of each class using Condition 1 [47], [48]:

$$\begin{aligned} Condition\ 1:\ & if\ (D^{MM}(\boldsymbol{r}^i_k) - D^{MM}(\boldsymbol{r}^i_{k-1}) > 0) \\ & and\ (D^{MM}(\boldsymbol{r}^i_k) - D^{MM}(\boldsymbol{r}^i_{k+1}) > 0) \\ & then\ (\{\boldsymbol{l}\}^i_{Q^i} \leftarrow \{\boldsymbol{l}\}^i_{Q^i} \cup \{\boldsymbol{r}^i_k\}) \end{aligned} \tag{6}$$

where $k = 2, 3, \ldots, L^i - 1$; Q^i is the number of local maxima of the ith class.

With the obtained local maxima, Voronoi tessellations are formed around each local maximum by attracting data samples of the same class to form a micro-cluster around it [47], [48]:

$$\mathbb{C}^i_{j^*} \leftarrow \mathbb{C}^i_{j^*} \cup \{\boldsymbol{x}^i_k\};\ j^* = \underset{j=1,2,\ldots,Q^i}{\operatorname{argmin}} (\parallel \boldsymbol{l}^i_j - \boldsymbol{x}^i_k \parallel^2); \tag{7}$$

where $\mathbb{C}^i_j$ is the micro-cluster formed around $\boldsymbol{l}^i_j$; $i = 1, 2, \ldots, C$ and $k = 1, 2, \ldots, K^i$. Centres, denoted as $\{\boldsymbol{q}\}^i_{Q^i}$ of the micro-clusters are then obtained as raw prototypes:

$$\boldsymbol{q}^i_j = \frac{1}{|\mathbb{C}^i_j|}\sum_{\boldsymbol{x}\in\mathbb{C}^i_j}\boldsymbol{x} \tag{8}$$

where $|\mathbb{C}^i_j|$ is the cardinality of $\mathbb{C}^i_j$.

Stage 2. Deriving the data-driven soft distance threshold

In the second stage, the average radius of area of influence around each micro-cluster centre, denoted as γ^i_G $(i = 1, 2, \ldots, C)$ is derived based on mutual distances of data samples and the level of granularity, denoted as G defined by users. G is a non-negative integer which can be determined without prior knowledge of the problems. γ^i_G provides an objective estimation of the distance between any two micro-clusters of the same class that are strongly associated with each other and can be combined as one. It is derived directly from data using Eqn. (9) [47], [48].

$$\begin{cases} \gamma^i_g = \frac{\sum_{j=1}^{K^i}\sum_{l=1}^{K^i}\|\boldsymbol{x}^i_j - \boldsymbol{x}^i_l\|^2}{(K^i)^2}, & if\ g = 1 \\ \gamma^i_g = \frac{\sum_{j=1}^{K^i}\sum_{l=1}^{K^i} v^i_{g,j,l}\|\boldsymbol{x}^i_j - \boldsymbol{x}^i_l\|^2}{\sum_{j=1}^{K^i}\sum_{l=1}^{K^i} v^i_{g,j,l}}, & else \end{cases} \tag{9}$$

where $v^i_{g,j,l} = \begin{cases} 1, & \parallel \boldsymbol{x}^i_j - \boldsymbol{x}^i_l \parallel^2 \leq \gamma^i_{g-1} \\ 0, & else \end{cases}$; $g = 1, 2, \ldots, G$.

Comparing with the commonly used crisp thresholds by other approaches, the average radius γ^i_G is a reliable soft distance threshold calculated directly from data and, thus, is always guaranteed to be meaningful. In addition, it also offers users more freedom to adjust the fineness of the learning outcomes without specialised expertise.

Stage 3. Identifying prototypes from local maxima

In the final stage of system identification, prototypes are identified as the more representative raw prototypes of each class. To identify such prototypes, the set of neighbouring raw prototypes, denoted by $\{\boldsymbol{q}^*\}^i_j$ around each individual raw prototype is firstly identified using Condition 2 with the help of the soft distance threshold, γ^i_G [47], [48]:

$$\begin{aligned} Condition\ 2:\ & if\ (\parallel \boldsymbol{q}^i_j - \boldsymbol{q}^i_k \parallel^2 \leq \gamma^i_G) \\ & then\ (\{\boldsymbol{q}^*\}^i_j \leftarrow \{\boldsymbol{q}^*\}^i_j \cup \{\boldsymbol{q}^i_k\}) \end{aligned} \tag{10}$$

where $i = 1, 2, \ldots, C$; $j, k = 1, 2, \ldots, Q^i$ and $j \neq k$.

More representative raw prototypes, $\{\hat{\boldsymbol{q}}\}^i_{N^i}$ (N^i is cardinality of $\{\hat{\boldsymbol{q}}\}^i_{N^i}$) of each class are then identified from $\{\boldsymbol{q}\}^i_{N^i}$ are performed using Condition 3 ($i = 1, 2, \ldots, C$; $j = 1, 2, \ldots, Q^i$) [47], [48]:

$$\begin{aligned} Condition\ 3:\ & if\ (D^{MM}(\boldsymbol{q}^i_j) > \max_{\boldsymbol{q}\in\{\boldsymbol{q}^*\}^i_j}(D^{MM}(\boldsymbol{q})) \\ & then\ (\{\hat{\boldsymbol{q}}\}^i_{N^i} \leftarrow \{\hat{\boldsymbol{q}}\}^i_{N^i} \cup \{\boldsymbol{q}^i_j\}) \end{aligned} \tag{11}$$

Finally, $\{\hat{\boldsymbol{q}}\}^i_{N^i}$ are used for forming Voronoi tessellations from data samples of the same class using Eqn. (12):

$$\mathbf{C}^i_{j^*} \leftarrow \mathbf{C}^i_{j^*} \cup \{\boldsymbol{x}^i_k\};\ j^* = \underset{j=1,2,\ldots,N^i}{\operatorname{argmax}} (\parallel \hat{\boldsymbol{q}}^i_j - \boldsymbol{x}^i_k \parallel^2) \tag{12}$$

and prototypes, $\{\boldsymbol{p}\}^i_{N^i}$ are obtained as ($i = 1, 2, \ldots, C$; $j = 1, 2, \ldots, N^i$):

$$\boldsymbol{p}^i_j = \frac{1}{|\mathbf{C}^i_j|}\sum_{\boldsymbol{x}\in\mathbf{C}^i_j}\boldsymbol{x} \tag{13}$$

With the extracted prototypes, $\{\boldsymbol{p}\}^i_{N^i}$ $(i = 1, 2, \ldots, C)$, the IF-THEN fuzzy rules, $\mathbf{R}^i$ $(i = 1, 2, \ldots, C)$ are built in the same form as Eqn. (1), and the system identification process is completed with the decision boundaries built from shape-free Voronoi tessellations formed around prototypes [50].

Remark 2: An important feature of SOIFS is its strong capability of handling potential class overlaps. As the prototype identification process is conducted class-wise, highly representative data samples of different classes will be identified from the overlapping regions as prototypes. Hence, more prototypes will reside in such regions compared with non-overlapping regions, and more precise decision boundaries will naturally be formed in these overlapping regions because of the finer partitioning of data locally.

The identification process of SOFIS is given by Algorithm 1 [47], [48].

B. Validation process

During the validation process, for each unlabelled data sample, $\boldsymbol{x}_k$, every IF-THEN fuzzy rule will provide a confidence score based on the similarity between $\boldsymbol{x}_k$ and the nearest prototype associated with the rule (one score per rule):

$$\lambda^i(\boldsymbol{x}_k) = \max_{\boldsymbol{p}\in\{\boldsymbol{p}\}^i_{N^i}} (e^{-\|\boldsymbol{x}_k - \boldsymbol{p}\|^2}) \tag{14}$$

Algorithm 1 SOFIS identification.

input: training set, $\{\boldsymbol{x}\}_K$
for $i = 1$ to C **do**
  calculate D^{MM} at $\{\boldsymbol{u}\}^i_{L^i}$ using Eqn. (4);
  rank $\{\boldsymbol{u}\}^i_{L^i}$ and obtain $\{\boldsymbol{r}\}^i_{L^i}$ using Eqn. (5);
  identify $\{\boldsymbol{l}\}^i_{Q^i}$ using Condition 1;
  form $\{\mathbb{C}\}^i_{Q^i}$ around $\{\boldsymbol{l}\}^i_{Q^i}$ using Eqn. (7);
  derive $\boldsymbol{q}^i_{Q^i}$ from $\{\mathbb{C}\}^i_{Q^i}$ using Eqn. (8);
  calculate γ^i_G from $\{\boldsymbol{x}\}^i_{K^i}$ using Eqn. (9);
  for $j = 1$ to Q^i **do**
    identify $\{\boldsymbol{q}^*\}^i_j$ using Condition 2;
  end for
  identify $\{\hat{\boldsymbol{q}}\}^i_{N^i}$ from $\{\boldsymbol{q}\}^i_{Q^i}$ using Condition 3;
  form $\{\mathbf{C}\}^i_{N^i}$ around $\{\hat{\boldsymbol{q}}\}^i_{N^i}$ using Eqn. (12);
  derive $\{\boldsymbol{p}\}^i_{N^i}$ from $\{\mathbf{C}\}^i_{N^i}$ using Eqn. (13);
end for
output: classifier, $\hat{y} = h(\boldsymbol{x})$

The class label of $\boldsymbol{x}_k$ is determined by the IF-THEN rule providing the highest confidence score, following the "winner takes all" principle [47], [48].

$$\hat{y}_k \leftarrow i^*;\ i^* = \underset{i=1,2,\ldots,C}{\arg\max}\,(\lambda^i(\boldsymbol{x}_k)) \tag{15}$$

Remark 3: SOFIS determines the class labels of unlabelled samples based on their similarity to the identified prototypes. Prototypes are highly representative samples in the data space preserving the structure and underlying pattern of the original data. Hence, the decision-making process of SOFIS can be traced easily by finding out the nearest prototypes in the data space, and the rationales behind its predictions can be interpreted by examining these prototypes.

B. AdaBoost

AdaBoost is one of the mostly used boosting algorithms [10], [15] aiming at solving binary classification problems (namely, $C = 2$). In the first learning cycle of AdaBoost, all training samples are given the equal weights. For the subsequent learning cycles, the weights are adjusted according to the classification results so that correctly classified training samples receive lower weights and incorrectly classified ones receive greater weights. AdaBoost focuses more on the training samples with higher weights since they are more challenging for the base classifiers. At the end, AdaBoost linearly combines all the base classifiers to create an ensemble model, where greater weights are given to the base classifiers with lower error rates.

The algorithmic procedure of AdaBoost is summarised by Algorithm 2 [10], [15].

C. Multi-Class AdaBoost Algorithm SAMME

SAMME [26] is a very popular and powerful multi-class boosting algorithm (namely, $C \geq 2$). SAMME modifies the original AdaBoost to multi-class classification without dividing the original problem into multiple binary ones.

Algorithm 2 AdaBoost.

input: training set, $\{\boldsymbol{x}\}_K$; number of iteration, T; base classifier, $\hat{y} = h(\boldsymbol{x})$
for $k = 1$ to K **do**
  initialise sample weight as: $w_{0,k} = \frac{1}{K}$;
end for
for $t = 1$ to T **do**
  train a base classifier $h_t(\boldsymbol{x})$ with the weighted $\{\boldsymbol{x}\}_K$;
  use $h_t(\boldsymbol{x})$ to predict the class labels $\{\hat{y}_t\}_K$ of $\{\boldsymbol{x}\}_K$;
  calculate the training error of $h_t(\boldsymbol{x})$ by Eqn. (16):

$$\varepsilon_t = \sum_{k=1}^{K} w_{t-1,k}\mathbb{I}(\hat{y}_{t,k} \neq y_k) \tag{16}$$

  if $\varepsilon_t < 0.5$ and $\varepsilon_t > 0$ **then**
    calculate classifier weight, α_t by Eqn. (17):

$$\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t}) \tag{17}$$

    for $k = 1$ to K **do**
      update sample weight, $w_{t,k}$ by Eqns. (18) and (19):

$$w_{t,k} = \frac{w_{t-1,k}e^{-\alpha_t(\mathbb{I}(\hat{y}_{t,k}=y_k)-\mathbb{I}(\hat{y}_{t,k}\neq y_k))}}{Z_t} \tag{18}$$

$$Z_t = \sum_{j=1}^{K} w_{t-1,j}e^{-\alpha_t(\mathbb{I}(\hat{y}_{t,j}=y_j)-\mathbb{I}(\hat{y}_{t,j}\neq y_j))} \tag{19}$$

    end for
  else
    $\alpha_t \leftarrow 0$;
    for $k = 1$ to K **do**
      $w_{t,k} \leftarrow w_{t-1,k}$;
    end for
  end if
end for
output: ensemble classifier,
$F_A(\boldsymbol{x}) = \underset{c=1,2,\ldots,C}{\arg\max}\,(\sum_{t=1}^{T}\alpha_t\mathbb{I}(\hat{y}_t = c))$

The algorithmic procedure of SAMME is summarised by Algorithm 3 [26], [31]. Comparing between Algorithms 2 and 3, one may conclude that the main differences between SAMME and AdaBoost include [31]: 1) the weights of base classifiers are updated differently (see Eqn. (20)), and; 2) the accuracy requirement of base classifiers is different. It is worth noting that SAMME is exactly the same as AdaBoost if $C = 2$.

III. Proposed FWAdaBoost Approach

As aforementioned, the vast majority of existing researches use mainstream classifiers, such as DT, SVM, KNN, ANN as the ensemble components for AdaBoost. The constructed ensemble models offer better classification precision than single-model classifiers, but they usually lack transparency and human-interpretability. To overcome this bottleneck, a novel multi-class AdaBoost-based fuzzy ensemble system is proposed. Technical details of this new ensemble system are described in this section.

Algorithm 3 SAMME.

input: training set, $\{\boldsymbol{x}\}_K$; number of iteration, T; base classifier, $\hat{y} = h(\boldsymbol{x})$
for $k = 1$ to K **do**
  initialise sample weight as: $w_{0,k} = \frac{1}{K}$;
end for
for $t = 1$ to T **do**
  train a base classifier $h_t(\boldsymbol{x})$ with the weighted $\{\boldsymbol{x}\}_K$;
  use $h_t(\boldsymbol{x})$ to predict the class labels $\{\hat{y}_t\}_K$ of $\{\boldsymbol{x}\}_K$;
  calculate the training error of $h_t(\boldsymbol{x})$ by Eqn. (16);
  if $\varepsilon_t < \frac{C-1}{C}$ and $\varepsilon_t > 0$ **then**
    calculate classifier weight, α_t by Eqn. (20):

$$\alpha_t = \frac{1}{2}(\ln(\frac{1-\varepsilon_t}{\varepsilon_t}) + \ln(C-1)) \tag{20}$$

    for $k = 1$ to K **do**
      update sample weight, $w_{t,k}$ by Eqns. (18) and (19);
    end for
  else
    $\alpha_t \leftarrow 0$;
    for $k = 1$ to K **do**
      $w_{t,k} \leftarrow w_{t-1,k}$;
    end for
  end if
end for
output: ensemble classifier,
$F_S(\boldsymbol{x}) = \underset{c=1,2,\ldots,C}{\arg\max} (\sum_{t=1}^{T} \alpha_t \mathbb{I}(\hat{y}_t = c))$

The general architecture of FWAdaBoost-SOFIES is given by Fig. 2, where one can see that the proposed ensemble system is composed of T SOFISs implemented in parallel. As stated in Section II.A, SOFIS determines the class label of a particular unlabelled sample based on the confidence scores produced by its IF-THEN fuzzy rules. These confidence scores are calculated based on the similarities between the unlabelled sample and prototypes of each class, providing valuable fuzzy information about the fitness of this unlabelled sample to the local models of data distribution. To utilise this information for better prediction precision, SAMME is modified such that the confidence scores are involved in both sample weight updating and ensemble output generation, resulting in the proposed FWAdaBoost algorithm. Hence, the constructed ensemble system is named as FWAdaBoost-SOFIES. Since SOFIS is employed as the base classifier and its technical details have been summarised in Section II.A, the main focus of this section is to present the proposed FWAdaBoost algorithm.

As aforementioned, FWAdaBoost follows the same framework of SAMME but uses the new 1) sample weight updating and 2) ensemble output generation schemes to better incorporate SOFIS for greater performance. The proposed new schemes are detailed as follows.

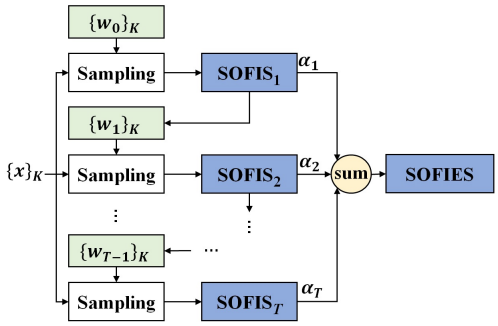

Fig. 2: Architecture of FWAdaboost-SOFIES.

A. The proposed sample weight updating scheme

During each iteration of FWAdaBoost, assuming the tth one, the sample weights are updated by Eqn. (21):

$$w_{t,k} = \frac{w_{t-1,k} e^{-\alpha_t \varphi_{t,k}}}{W_t} \tag{21}$$

where

$$\varphi_{t,k} = \lambda_t^{y_k}(\boldsymbol{x}_k) - \lambda_t^{c^*}(\boldsymbol{x}_k) \tag{22}$$

$$W_t = \sum_{j=1}^{K} w_{t-1,j} e^{-\alpha_t \varphi_{t,j}} \tag{23}$$

$$\lambda_t^{c^*}(\boldsymbol{x}_k) = \max_{\substack{c=1,2,\ldots,C; \\ c \neq y_k}} (\lambda_t^c(\boldsymbol{x}_k)) \tag{24}$$

Here, W_t (Eqn. (23)) is a normalisation factor; $\lambda_r^{y_k}(\boldsymbol{x}_k)$ is the confidence score produced by the fuzzy rule $\mathbf{R}_t^{y_k}$ of the tth base classifier, which is identified from training samples of the same class as $\boldsymbol{x}_k$; $\lambda_t^{c^*}(\boldsymbol{x}_k)$ (Eqn. (24)) is the maximum confidence score produced by other fuzzy rules of the tth base classifier. Consequently, $\varphi_{t,k}$ can be reformulated as:

$$\varphi_{t,k} = |\varphi_{t,k}|(\mathbb{I}(\hat{y}_{t,k} = y_k) - \mathbb{I}(\hat{y}_{t,k} \neq y_k)) \tag{25}$$

where a greater value of $|\varphi_{t,k}|$ suggests that SOFIS has higher confidence towards its prediction, and vice versa. By substituting Eqn. (25) into Eqn. (21), Eqn. (21) can be converted into a similar form as Eqn. (18).

Generally speaking, predictions made by the base classifier, namely, SOFIS can be roughly divided into four categories (note that this is not hardcoded):

Category 1: correct predictions with high confidence, namely, $\lambda_r^{y_k}(\boldsymbol{x}_k)$ is significantly higher than $\lambda_t^{c^*}(\boldsymbol{x}_k)$;

Category 2: correct predictions with low confidence, namely, $\lambda_r^{y_k}(\boldsymbol{x}_k)$ is slightly higher than $\lambda_t^{c^*}(\boldsymbol{x}_k)$;

Category 3: wrong predictions with high confidence, namely, $\lambda_r^{y_k}(\boldsymbol{x}_k)$ is slightly lower than $\lambda_t^{c^*}(\boldsymbol{x}_k)$, and;

Category 4: wrong predictions with low confidence, namely, $\lambda_r^{y_k}(\boldsymbol{x}_k)$ is significantly lower than $\lambda_t^{c^*}(\boldsymbol{x}_k)$.

Hence, the value of $\varphi_{t,k}$ calculated by Eqn. (22) serves as a soft indicator describing both the correctness of the prediction on the class label of $\boldsymbol{x}_k$ and the level of confidence SOFIS has towards its prediction.

In contrast with the conventional sample weight updating scheme used by AdaBoost and SAMME, involving the soft

indicator $\varphi_{t,k}$ in sample weight updating offers several benefits. Firstly, more weights are put on samples that are classified by the base classifier wrongly with high confidence (namely, category 4) and less weights are given to the samples classified correctly with high confidence (namely, category 1). Thus, in each iteration, the base classifier is able to focus more on these highly challenging samples and pay less attention to these easy-to-classify samples. Secondly, weights of correctly classified samples but with low confidence (namely, category 2) will decrease more slowly. These samples are much closer to the classification boundaries (with low classification margins) and could be misclassified if the boundaries are altered. Slowing down the weight decline for these data samples enables the base classifier to pay sufficient attention to these samples in later iterations, effectively preventing the base classifier from making wrong predictions on them. Thirdly, the absolute value of $\varphi_{t,k}$ is always changing smoothly. This can effectively avoid the abrupt increase/decrease in the sample weights, preventing overfitting. Lastly, these confidence scores are produced by base classifiers based on the similarity between data samples and prototypes, and are intrinsically explainable to/by human. Hence, involving confidence scores in sample weight updating can further improve the interpretability of the boosting process because one can easily understand the rationales behind the changes of sample weights.

B. The proposed ensemble output generation scheme

The confidence scores that the base classifiers produced on unlabelled testing samples are also involved in the ensemble output generation. For a particular testing sample, $\boldsymbol{x}_k$, each base classifier (assuming the tth one) will produce a predicted class label denoted as $\hat{y}_{t,k}$. The predicted label, $\hat{y}_{t,k}$ is further encoded as a C-dimensional vector, $\hat{\boldsymbol{Y}}_{t,k} = [\hat{Y}^1_{t,k}, \hat{Y}^2_{t,k}, \ldots, \hat{Y}^C_{t,k}]^T$ using Eqn. (26) [26], where $c = 1, 2, \ldots, C$.

$$\hat{Y}^c_{t,k} = \begin{cases} 1, & if\ c = \hat{y}_{t,k} \\ -\frac{1}{C-1}, & else \end{cases} \tag{26}$$

The final output, namely, the predicted label of $\boldsymbol{x}_k$ of the ensemble system is produced by Eqn. (27) as follows.

$$\hat{y}_k = F(\boldsymbol{x}_k) = \underset{c=1,2,\ldots,C}{\arg\max} (f^c(\boldsymbol{x}_k)) \tag{27}$$

where

$$f^c(\boldsymbol{x}_k) = \sum_{t=1}^{T} \alpha_t \hat{\varphi}_{t,k} \hat{Y}^c_{t,k} \tag{28}$$

$$\hat{\varphi}_{t,k} = \lambda_t^{\hat{y}_{t,k}}(\boldsymbol{x}_k) - \lambda_t^{\hat{c}^*}(\boldsymbol{x}_k) \tag{29}$$

$$\lambda_t^{\hat{c}^*}(\boldsymbol{x}_k) = \max_{\substack{c=1,2,\ldots,C; \\ c \neq \hat{y}_k}} (\lambda_t^c(\boldsymbol{x}_k)) \tag{30}$$

Here $\hat{\varphi}_{t,k}$ is the difference between the highest and the second highest confidence scores, indicating the level of confidence the base classifier has towards its prediction. A higher value of $\hat{\varphi}_{t,k}$ means that the base classifier is highly confident about the correctness of its prediction, and vice versa. In this way, the predictions that base classifiers are highly confident with can contribute more in the final ensemble outputs and, at the same time, the predictions with less confidence will play a much smaller role in the final outputs, resulting in greater overall prediction precision of the ensemble system.

Remark 4: FWAdaBoost only utilise two of the C confidence scores in sample weight updating and ensemble output generation, hence, it can be used for both binary and multi-class classification problems.

The algorithmic procedure of FWAdaBoost is summarised by Algorithm 4.

Algorithm 4 FWAdaBoost.

input: training set, $\{\boldsymbol{x}\}_K$; number of iteration, T;
base classifier, $\hat{y} = h(\boldsymbol{x})$
for $k = 1$ to K **do**
initialise sample weight as: $w_{0,k} = \frac{1}{K}$;
end for
for $t = 1$ to T **do**
train a base classifier $h_t(\boldsymbol{x})$ with the weighted $\{\boldsymbol{x}\}_K$;
use $h_t(\boldsymbol{x})$ to predict the class labels $\{\hat{y}_t\}_K$ of $\{\boldsymbol{x}\}_K$;
calculate the training error of $h_t(\boldsymbol{x})$ by Eqn. (16);
if $\varepsilon_t < \frac{C-1}{C}$ and $\varepsilon_t > 0$ **then**
calculate classifier weight, α_t by Eqn. (20);
for $k = 1$ to K **do**
update sample weight, $w_{t,k}$ by Eqn. (21);
end for
else
$\alpha_t \leftarrow 0$;
for $k = 1$ to K **do**
$w_{t,k} \leftarrow w_{t-1,k}$;
end for
end if
end for
output: ensemble classifier,
$F(\boldsymbol{x}) = \underset{c=1,2,\ldots,C}{\arg\max} (\sum_{t=1}^{T} \alpha_t \hat{\varphi}_t \hat{Y}^c_t)$

IV. Theoretical Justification

In this section, a theoretical analysis on the upper bound of the training error of FWAdaBoost is presented. It is acknowledged that the mathematical derivations to the proof are inspired by [15], [26]

Theorem 1: *The following bound holds on the training error, e_o of the ensemble system constructed by FWAdaBoost:*

$$e_o < \prod_{t=1}^{T} W_t \tag{31}$$

Proof: For a particular training sample $\boldsymbol{x}_k$, if the predicted label by Eqn. (27), $\hat{y}_k$ does not match its true label y_k, namely, $\mathbb{I}(\hat{y}_k \neq y_k) = 1$, the following inequality holds:

$$L(\boldsymbol{x}_k) = f^{y_k}(\boldsymbol{x}_k) - f^{\hat{y}_k}(\boldsymbol{x}_k) < 0 \tag{32}$$

$L(\boldsymbol{x}_k)$ in inequality (32) can be reformulated as:

$$L(\boldsymbol{x}_k) = \sum_{t=1}^{T} \alpha_t \hat{\varphi}_{t,k} \zeta_{t,k} \tag{33}$$

where $\zeta_{t,k} = \hat{Y}_{t,k}^{y_k} - \hat{Y}_{t,k}^{\hat{y}_k}$ and, based on Eqn. (26), there is (on condition that $\hat{y}_k \neq y_k$):

$$\zeta_{t,k} = \begin{cases} \frac{C}{C-1}, & if\ \hat{y}_{t,k} = y_k\ \&\ \hat{y}_{t,k} \neq \hat{y}_k \\ -\frac{C}{C-1}, & if\ \hat{y}_{t,k} = \hat{y}_k\ \&\ \hat{y}_{t,k} \neq y_k \\ 0, & if\ \hat{y}_{t,k} \neq y_k\ \&\ \hat{y}_{t,k} \neq \hat{y}_k \end{cases} \tag{34}$$

The following relationship can be derived via a comparison between Eqns. (22) and (29),

$$\begin{cases} |\varphi_{t,k}| = \hat{\varphi}_{t,k}, & if\ \hat{y}_{t,k} = y_k \\ |\varphi_{t,k}| \geq \hat{\varphi}_{t,k}, & if\ \hat{y}_{t,k} \neq y_k \end{cases} \tag{35}$$

Based on Eqns. (34) and (35), one can tell that the following inequality always holds:

$$\frac{C}{C-1}\varphi_{t,k} \leq \hat{\varphi}_{t,k}\zeta_{t,k} \tag{36}$$

Considering that $\alpha_t \geq 0\ \forall t = 1, 2, ..., T$, inequality (37) can be derived from inequalities (32) and (36):

$$\frac{C}{C-1}\sum_{t=1}^{T}\alpha_t\varphi_{t,k} < \sum_{t=1}^{T}\alpha_t\hat{\varphi}_{t,k}\zeta_{t,k} < 0 \tag{37}$$

Hence, there is $\sum_{t=1}^{T}\alpha_t\varphi_{t,k} < 0$. By unravelling the proposed sample weight updating rule (Eqn. (21)), there is [15]:

$$w_{T,k} = \frac{e^{-\sum_{t=1}^{T}\alpha_t\varphi_{t,k}}}{K\prod_{t=1}^{T}W_t} \tag{38}$$

and the following inequality (39) holds:

$$\mathbb{I}(\hat{y}_k \neq y_k) < e^{-\sum_{t=1}^{T}\alpha_t\varphi_{t,k}} = w_{T,k}\cdot K\prod_{t=1}^{T}W_t \tag{39}$$

Since the overall training error of the ensemble system, e_o can be formulated in the form of Eqn. (40) [15],

$$e_o = \frac{1}{K}\sum_{k=1}^{K}\mathbb{I}(\hat{y}_k \neq y_k) \tag{40}$$

and inequality (41) can be obtained.

$$e_o < \frac{1}{K}\sum_{k=1}^{K}e^{-\sum_{t=1}^{T}\alpha_t\varphi_{t,k}} = \prod_{t=1}^{T}W_t \tag{41}$$

■

Remark 5: It can be concluded from inequality (41) that the training error of FWAdaBoost is upper bounded. This conclusion coincides with the Theorem 1 in [15], but it considers the more general multi-class classification problems.

V. Experimental Investigations

In this section, numerical examples on a wide range of benchmark problems are presented to demonstrate the efficacy of the proposal FWAdaBoost-SOFIES. The algorithms were developed on MATLAB 2018b platform. The performance was evaluated on a laptop with dual core i7 processer 2.60GHz×2 and 16GB RAM. By default, all the numerical experiments were conducted in offline scenarios and the reported results were obtained as the average of 25 Monte-Carlo experiments to allow a certain degree of randomness.

Since FWAdaBoost is suitable for both binary and multi-class classification problems, benchmark datasets of the two types are used for numerical experiments. In this study, nine numerical binary classification problems, 12 numerical multi-class classification problems and two multi-class image classification problems are considered. Key information of binary problems is tabulated in Supplementary Table S1. Key information of the 14 numerical and image classification problems is provided in Supplementary Table S2. Web links to the 23 datasets are given in Supplementary Table S3.

A. Visual illustration

In this subsection, an illustrative example is presented to demonstrate the process of FWAdaBoost-SOFIES to build a more precise decision-boundaries from training data. For visual clarity, principle component analysis (PCA) is applied to reduce the dimensionality of the original data to two and the obtained first two principle components (PCs) are further normalised to the value range of $[0, 1]$. After pre-processing, half of the dataset is used to form the training set and the rest is used for testing. In this example, Euclidean distance is used as the distance measure for convenient illustration. The constructed ensemble system by FWAdaBoost is composed of six base classifiers, and the level of granularity, G is set as 3.

Firstly, the spatial distributions of the 2-dimensional training and testing samples in the data space are depicted in Supplementary Fig. S1(a) and (b), respectively, where dots "·" in light blue represent data samples of class 1 and the dots "·" in orange represent data samples of class 2. It can be observed from Supplementary Fig. S1 that the data samples of two classes are mixed together in the dense central area.

Same as the standard AdaBoost-based ensemble framework, the six base classifiers of FWAdaBoost-SOFIES are learned sequentially from training data, and the classification boundaries of each base classifier will be different due to the weighted random sampling. The prototypes identified from training data by SOFIS during each training iteration are visualised in Supplementary Fig. S2(a)-(f), where the larger dots "•" in blue represent the prototypes identified from data samples of class 1 and the larger dots "•" in red represent the prototypes identified from data samples of class 2. The Voronoi tessellations [50] formed by these prototypes are also visualised in Supplementary Fig. S2, where the black dash lines "- -" represent the boundaries of Voronoi tessellations. Based on the formed Voronoi tessellations in the data space, the classification boundaries of the predictive model are then built naturally, presented by the green lines "—" in Supplementary Fig. S2. The integrated classification boundaries of the six base classifiers are given by Fig. 3. By comparing between Supplementary Fig. S2 and Fig. 3, one can see that more sophisticated classification boundaries are constructed by forming an ensemble system from the six base learners.

To demonstrate the effectiveness of the proposed FWAdaBoost algorithm, the classification error rate of FWAdaBoost-SOFIES on the testing set is reported in Supplementary Table S4, and the respective classification error rates of the six base classifiers are reported in the same table as well. One can

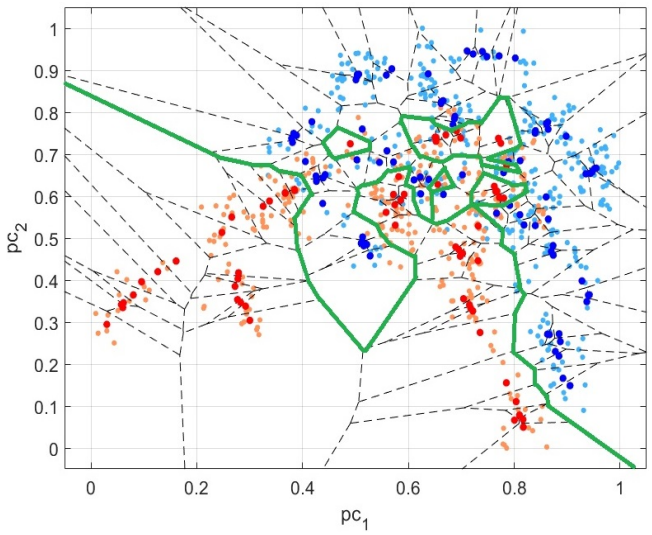

Fig. 3: The integrated decision boundaries of the six base learners through training (dots"·" in light blue –data samples of class 1; the dots "·" in orange – data samples of class 2; the larger dots"•" in blue - prototypes of class 1; the larger dots "•" in red – prototypes of class 2; black dash lines "- -" – the boundaries of Voronoi tessellations; the green lines "—" – the decision boundaries)

clearly see from Supplementary Table S3 that FWAdaBoost is capable of constructing a more precise predictive ensemble model from the learned base classifiers.

B. Sensitivity analysis

In this subsection, numerical experiments are conducted to investigate the influence of the level of granularity, G and the number of base classifiers, T within the ensemble system on the performance of FWAdaBoost-SOFIES. Four benchmark datasets (one binary ones and three multi-class ones) are used for the numerical examples presented in this subsection, namely, MG, IS, MF and WF. For MG, MF, and WF datasets, half of the data samples are randomly selected out to form the training sets, and the remaining ones are used to form the testing sets. For IS dataset, the original train/test split is kept, but the orders of the training samples are scrambled randomly during the experiments.

Firstly, the influence of the externally controlled parameter, G on system performance is studied. In this numerical example, the ensemble system is composed of 20 base classifiers, namely, $T = 20$. The value of G varied from 1 to 14. The obtained results in terms of classification accuracy rates on the testing sets are reported in Supplementary Table S5, where the prediction results by SOFIS are also reported as the baseline.

It can be observed from Supplementary Table S5 that the proposed FWAdaBoost algorithm effectively boosts the prediction performance of SOFIS, and the boosting effect is even more significant when a lower level of granularity is used. Taking MG and WF datasets as example, the prediction accuracy rates of FWAdaBoost-SOFIES are improved by approximately 30% and 13% with the level of granularity set as $G = 1$, and approximately 14% and 10% with the level of granularity set as $G = 2$. The proposed ensemble system reaches the maximum prediction precision with G greater than 5 on the considered benchmark problems. Beyond this range, increasing the value of G will not bring significant improvement to the system performance. Meanwhile, SOFIS reaches the maximum prediction precision with G greater than 7. This example also serves as a strong evidence, showing the effectiveness of the proposed FWAdaBoost on boosting the performance of SOIFS.

Secondly, the influence of the externally controlled parameter, T on system performance is studied. In this numerical example, the number of the base classifiers in the ensemble system, namely, T varies from 5 to 35. Two different levels of granularity, G are considered, namely, $G = 2$ and $G = 12$. The obtained results in terms of classification accuracy rates on the testing sets are reported in Supplementary Table S6. The prediction results by SOFIS under the same experimental settings (in this case, $T = 1$) are also reported in the same table for better illustration.

Supplementary Table S6 shows that generally, the more base classifiers are used in the ensemble system, the better the prediction accuracy will be. However, it is also worth noting that when FWAdaBoost-SOFIES is composed of more than 20 base classifiers, adding more base classifiers to the ensemble systems will not bring significant benefits to the overall prediction performance.

Based on Supplementary Tables S5 and S6, for the rest of this section, unless specifically declared otherwise, the level of granularity of SOFIS is set to be $G = 12$ to ensure that SOFIS always achieves strong prediction performance, and the ensemble classifier is composed of 20 base classifiers, namely, $T = 20$. Note that this is a recommended parameter setting, and the performance of FWAdaBoost-SOFIES can be further improved on a particular problem by tuning the two parameters. However, it is demonstrated through numerical examples later that with this recommended setting, FWAdaBoost-SOFIES outperforms its competitors on a wide range of benchmark problems (different from the four datasets considered in this subsection).

C. Ablation analysis

In this subsection, ablation analysis is performed to justify the effectiveness and validity of the proposed concept and general principles. As stated in Section III, FWAdaBoost employs the same framework of the widely used SAMME, but its sample weight updating and ensemble output generation schemes have been modified to utilise the confidence scores produced by SOFIS during decision-making for greater prediction precision. In the following example, two variations of FWAdaBoost are considered, namely, FWAdaBoost1 and FWAdaBoost2. Both variations follow the same framework of SAMME. However, FWAdaBoost1 uses the proposed sample weight updating scheme only, while FWAdaBoost2 uses the proposed ensemble output generation scheme only. 10 benchmark datasets are used in this example, which include four binary ones, namely, MG, PID, PW and SP, and six multi-class ones, namely, CA, IS, MF, OR, PR and WF. For MG,

PID, SP, CA, MF and WF datasets, half of the data samples are randomly selected out to form the training sets, and the remaining ones are used to form the testing sets. For IS, OR and PR datasets, the original train/test splits are kept, but the orders of the training samples are scrambled randomly during the experiments. The prediction results of the ensemble systems constructed by FWAdaBoost, FWAdaBoost1 and FWAdaBoost2 on the testing sets of the eight benchmark problems are tabulated in Supplementary Table S7 with the results obtained by SAMME given as the baseline. The average classification accuracy rates over the ten datasets are presented by Fig. 4.

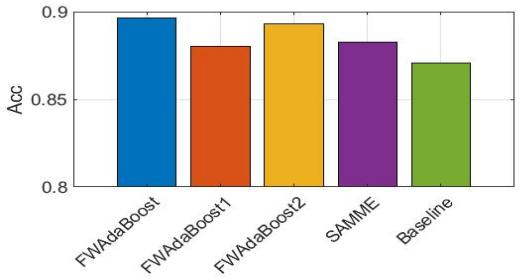

Fig. 4: Overall average classification accuracy comparison for ablation analysis

One can see from Supplementary Table S7 and Fig. 4 that FWAdaBoost2 is able to construct more precise ensemble systems than SAMME in most of the cases. However, FWAdaBoost1 only outperforms SAMME (AdaBoost) on binary classification problems. This is due to the fact that FWAdaBoost1 gives much less weights to correct predictions with high confidences during the training process than SAMME. As a result, the classification boundaries of some base classifiers within the ensemble system are altered by the less confident predictions and wrong predictions to a greater degree in order to correctly classifying such samples. However, this may also lead to the issue that some of the easy-to-classify samples will be wrongly classified by these base classifiers. This causes the lower classification performance of the ensemble system constructed by FWAdaBoost1. However, with both proposed sample weight updating and ensemble output generation schemes, this issue can be successfully addressed because the levels of confidence each base classifier have towards its individual predictions are considered in the overall ensemble outputs. Therefore, it can be concluded from Supplementary Table S7 and Fig. 4 that FWAdaBoost can effectively improve the prediction performance of the overall ensemble system.

D. Performance demonstration

In this subsection, numerical experiments are conducted for comparing the performance of FWAdaBoost over AdaBoost and its variations. Since FWAdaBoost is suitable for both binary and multi-class classification problems, experiments are performed on the two types of problems separately.

Firstly, seven binary classification problems, namely, ESR, GC, OD, PID, PW, SE and SP are used for evaluating the performance of FWAdaBoost-SOFIES. During the experiments, the original train/test split of OD dataset is kept but the order of the training samples are randomly scrambled in each experiment. For the other six datasets, 50% of the data samples are selected out randomly to form the training sets and the rest are used for testing. In addition, the following six two-class boosting algorithms are used to construct ensemble classifiers with SOFIS serving as benchmark comparison: 1) AdaBoost [10]; 2) ReAdaBoost [15], [17]; 3) AveBoost2 [20]; 4) PAdaBoost [18]; 5) NDAdaBoost [22], and; 6) RoAdaBoost [12]. Classification results of the ensemble systems constructed by FWAdaBoost and its competitors are reported in Table I in terms of balanced accuracy rate ($BAcc$) [56], where the best results are highlighted. The results obtained by SOFIS are also reported in the same table as the baseline. For better demonstration, performances of the eight algorithms over the seven binary classification datasets are ranked from best to worst in terms of $BAcc$, and the average ranks are reported in Table I. It is shown by Table I that the ensemble system constructed by FWAdaBoost is able to achieve very high balanced accuracy on the seven binary datasets, surpassing the alternative ensemble systems constructed by other two-class boosting algorithms involved in this example. This example justifies the effectiveness of the proposed algorithm.

Next, nine multi-class benchmark datasets are used for performance evaluation, which include AB, CA, GP, LR, OR, PB, PR, SH and TE. Following the same experimental protocol used before, for OR and PR datasets, the original train/test splits are kept, but the orders of the training samples are scrambled randomly in each experiment. For the other seven datasets, 50% of the data samples are selected out randomly to build the training sets and the remaining samples are used to form the testing sets. The following four well-known multi-class boosting algorithms are used for benchmark comparison: 1) AdaBoost.M1 [24]; 2) AdaBoost.M2 [24]; 3) SAMME [26], and; 4) ReSAMME [26].

Classification performances of the ensemble classifiers constructed by the five multi-class boosting algorithms are reported in Table II in terms of $BAcc$ [56] with the best results being highlighted. The results obtained by SOFIS are also reported in the same table as baseline. Similar, average performance ranks of the six approaches over the nine datasets are also reported. Table II shows that the ensemble system constructed by the proposed FWAdaBoost algorithm outperforms alternative ensemble systems with the highest overall rank, demonstrating its efficacy.

It is also interesting to notice from the comparison given by Tables I and II that FWAdaBoost, ReAdaBoost and ReSAMME all involve the prediction confidences (confidence scores in this case) in sample weight updating and ensemble output generation, but FWAdaBoost outperforms the other two algorithms in most of the cases thanks to the unique operating mechanism proposed in this paper.

Then, the performance of FWAdaBoost-SOFIES is compared with a number of classification approaches on the nine multi-class datasets for benchmark comparison. The following nine widely used classification approaches are used for comparison (the first three are zero-order FISs): 1) self-organizing fuzzy inference ensemble system (SOFEnsemble)

TABLE I
PERFORMANCE COMPARISON BETWEEN DIFFERENT BOOSTING ALGORITHMS ON BINARY CLASSIFICATION PROBLEMS

Algorithm	ESR	GC	OD	PID	PW	SE	SP	Rank
FWAdaBoost	0.8296	**0.6044**	0.9439	**0.6179**	**0.9423**	0.5003	0.8342	**2.5**
AdaBoost	0.8112	0.5890	0.9398	0.6126	0.9367	**0.5269**	0.8099	6.2
ReAdaBoost	0.8273	0.5972	**0.9440**	0.6117	0.9419	0.5004	0.8337	4.4
AveBoost2	0.8213	0.5965	0.9452	0.6128	0.9413	0.5106	0.8283	4.3
PAdaBoost	0.8246	0.6034	0.9435	0.6158	0.9402	0.5002	0.8342	4.2
NDAdaBoost	**0.8298**	0.6015	0.9421	0.6118	0.9399	0.5048	0.8317	4.6
RoAdaBoost	0.8277	0.6036	0.9423	0.6155	0.9422	0.5042	**0.8349**	3.0
Baseline	0.8112	0.5863	0.9398	0.5994	0.9367	0.5269	0.8099	6.8

TABLE II
OVERALL PERFORMANCE COMPARISON BETWEEN DIFFERENT BOOSTING ALGORITHMS ON MULTI-CLASS CLASSIFICATION PROBLEMS

Algorithm	AB	CA	GP	LR	OR	PB	PR	SH	TE	Rank
FWAdaBoost	0.5398	**0.7389**	**0.7378**	**0.9409**	0.9777	0.6650	0.9763	**0.9078**	**0.9886**	**2.1**
AdaBoost.M1	0.5083	0.7288	0.7360	0.9287	0.9759	**0.7084**	0.9757	0.8947	0.9873	4.0
AdaBoost.M2	0.5325	0.7202	0.6869	0.9346	**0.9791**	0.6855	**0.9779**	0.9036	0.9884	3.1
SAMME	0.5189	0.7287	0.7340	0.8924	0.9759	**0.7084**	0.9757	0.8775	0.9851	4.8
ReSAMME	**0.5416**	0.7094	0.7706	0.9361	0.9790	0.6593	0.9773	0.9036	0.9882	2.8
Baseline	0.5080	0.7287	0.7360	0.9287	0.9760	**0.7084**	0.9757	0.8945	0.9873	4.2

[53]; 2) zero-order autonomous learning multiple-model classifier (ALMMo0) [54]; 3) eClass0 classifier [39]; 4) SVM with linear kernel; 5) KNN; 6) DT; 7) multilayer perceptron (MLP); 8) sequence classifier (SC) [58], and; 9) extreme learning machine (ELM) [59].

In this example, SOFEnsemble, ALMMo0, eClass0 and SC classifiers follow the recommended parameter settings given by [39], [53], [54], [58]. For SVM, KNN, MLP, DT and ELM classifiers, five different parameter settings are considered for each of them during the experiments and the best performances are reported. In particular, the box constraint, C of SVM is set as: $C \in \{0.5, 0.75, 1.0, 1.25, 1.5\}$. The number of nearest neighbours, k for KNN is set as: $k \in \{6, 8, 10, 12, 14\}$. Five different architectures are considered for MLP, namely, *i)* one hidden layer with 20 neurons; *ii)* two hidden layers with 20 neurons in each; *iii)* three hidden layers with 20 neurons in each; *iv)* one hidden layer with 40 neurons, and; *v)* two hidden layers with 40 neurons in each. The maximum depth of DT varies from 25, 50, 100, 150 and 200. The maximum number of neurons of ELM varies from 100, 150, 200, 250 and 300.

In addition, the following eight ensemble classifiers based on AdaBoost.M2 and SAMME are created from SVM, KNN, DT and MLP with the respective best performing parameter settings, and used for benchmark comparison: 1) AdaBoost.M2-based SVM ensemble classifier (AdaBoost.M2-SVM); 2) AdaBoost.M2-based KNN ensemble classifier (AdaBoost.M2-KNN); 3) AdaBoost.M2-based DT ensemble classifier (AdaBoost.M2-DT); 4) AdaBoost.M2-based MLP ensemble classifier (AdaBoost.M2-MLP); 5) SAMME-based SVM ensemble classifier (SAMME-SVM); 6) SAMME-based KNN ensemble classifier (SAMME-KNN); 7) SAMME-based DT ensemble classifier (SAMME-DT), and; 8) SAMME-based MLP ensemble classifier (SAMME-MLP). Following the experimental setting of FWAdaBoost-SOFIES, the number of base classifiers for each ensemble model is set as 20.

The results obtained by FWAdaBoost-SOFIES and its competitors are reported in Table III in terms of $BAcc$ [56]. The results obtained by SOFIS are also presented in the same table as baseline. The best results in Table III are in bold. For visual clarity, the performances of the 18 algorithms involved in this numerical example are ranked from best to worst in terms of $BAcc$ per dataset, and the average ranks are also tabulated in Table III. From Table III one can see that the proposed FWAdaBoost-SOFIES is ranked the top place on nine multi-class classification problems, demonstrating the efficacy of the proposed ensemble approach for classification.

To examine the statistical significance of the better performance achieved by FWAdaBoost-SOFIES, over the 18 single-model and ensemble competitors and on the nine benchmark problems, pairwise Wilcoxon signed rank tests [57] are conducted. The outcomes of the pairwise tests in terms of p-value are tabulated in Supplementary Table S8, where the cascaded classification results by each approach from the 25 experiments are used. It can be observed that 88.9% of the p-values returned by the pairwise Wilcoxon tests are below the level of significance specified by $\alpha = 0.05$. This suggests that the performance of FWAdaBoost-SOFIES is significantly better than the other 18 classifiers.

As the number and model complexity of base classifiers can have great impact on the performances of the resulting ensemble models, additional numerical experiments are performed to compare the performances between FWAdaBoost-SOFIES and the eight AdaBoost.M2-based and SAMME-based ensemble classifiers used in the previous example with the respective optimised parameter settings for better evaluation. AdaBoost.M2-SOFIES and SAMME-SOFIES are also involved in the numerical comparison. In this example, the nine multi-class benchmark datasets used before are employed,

TABLE III
PERFORMANCE COMPARISON BETWEEN DIFFERENT SINGLE-MODEL AND ENSEMBLE CLASSIFIERS

Algorithm	AB	CA	GP	LR	OR	PB	PR	SH	TE	Rank
FWAdaBoost-SOFIES	0.5398	0.7389	0.7378	**0.9409**	0.9777	0.6650	**0.9763**	0.9078	0.9886	**4.6**
SOFIS	0.5080	0.7287	0.7360	0.9287	0.9760	0.7084	0.9757	0.8945	0.9873	7.4
SOFEnsemble	0.5373	0.6775	0.7440	0.9370	0.9100	0.5415	0.9426	0.8984	0.9799	10.6
ALMMo0	0.4878	0.7100	0.6516	0.9198	0.9776	0.6875	0.9750	0.8953	0.9857	8.4
eClass0	0.4765	0.6784	0.2971	0.5074	0.8727	0.6440	0.8473	0.7000	0.7783	17.0
SVM	0.5259	0.7431	0.4207	0.8524	0.9625	0.5565	0.9563	**0.9173**	**0.9906**	9.3
KNN	0.5337	0.7116	0.7530	0.9294	**0.9782**	0.6040	0.9760	0.8721	0.9754	7.7
DT	0.5332	0.8338	0.7034	0.6848	0.8542	0.7752	0.9121	0.6943	0.9030	12.6
MLP	0.5503	0.7770	0.5356	0.6535	0.9502	0.5763	0.9573	0.6586	0.9800	11.7
SC	0.4640	0.7723	**0.8106**	0.8557	0.9547	0.6461	0.9516	0.8955	0.9812	8.8
ELM	0.3750	0.6326	0.4752	0.5606	0.9336	0.2009	0.9595	0.4515	0.9802	15.6
AdaBoost.M2-SVM	0.5205	0.7393	0.4175	0.8522	0.9620	0.6273	0.9577	0.9119	0.9882	9.6
AdaBoost.M2-KNN	0.5223	0.7092	0.7383	0.9254	0.9773	0.5861	0.9761	0.8595	0.9762	9.4
AdaBoost.M2-DT	0.5339	0.8349	0.7550	0.7447	0.9206	**0.7861**	0.9409	0.7743	0.9466	9.3
AdaBoost.M2-MLP	**0.5515**	0.7804	0.5457	0.7072	0.9642	0.6237	0.9643	0.8140	0.9849	8.3
SAMME-SVM	0.5259	0.7431	0.4207	0.8354	0.9572	0.6713	0.9562	0.8888	0.9897	9.5
SAMME-KNN	0.5337	0.7116	0.7530	0.9294	**0.9782**	0.6040	0.9760	0.8721	0.9754	7.7
SAMME-DT	0.5435	**0.8340**	0.7280	0.7522	0.8787	0.7752	0.9189	0.7338	0.9052	10.5
SAMME-MLP	0.5514	0.7659	0.5354	0.6994	0.9504	0.5768	0.9546	0.7305	0.9710	12.0

TABLE IV
PERFORMANCE COMPARISON BETWEEN DIFFERENT ENSEMBLE CLASSIFIERS WITH OPTIMISED PARAMETER SETTINGS

Algorithm	AB	CA	GP	LR	OR	PB	PR	SH	TE	MN	FMN	Rank
FWAdaBoost-SOFIES	0.5345	0.7289	0.7049	**0.9282**	0.9753	0.6521	**0.9759**	0.8909	0.9863	**0.9594**	**0.8447**	**2.9**
AdaBoost.M2-SOFIES	0.5305	0.7171	0.6840	0.9190	**0.9758**	0.6798	0.9757	0.8881	0.9857	0.9579	0.8408	3.8
AdaBoost.M2-SVM	0.5225	0.7369	0.4140	0.8475	0.9590	0.6051	0.9551	**0.9094**	0.9859	0.9300	0.8359	6.4
AdaBoost.M2-KNN	0.5212	0.6801	0.7010	0.9060	0.9737	0.5562	0.9742	0.8461	0.9697	0.9432	0.8114	7.0
AdaBoost.M2-DT	0.5254	**0.8398**	**0.7467**	0.7622	0.9186	**0.8093**	0.9396	0.7694	0.9411	0.8942	0.8173	6.5
AdaBoost.M2-MLP	**0.5600**	0.7511	0.5529	0.7088	0.9661	0.6208	0.9654	0.7996	**0.9888**	0.5428	0.7135	6.5
SAMME-SOFIES	0.5205	0.7151	0.6986	0.9142	0.9730	0.6936	0.9736	0.8712	0.9834	0.9488	0.8280	5.5
SAMME-SVM	0.5268	0.7421	0.4189	0.8487	0.9543	0.6774	0.9535	0.8889	0.9883	0.9266	0.8333	6.0
SAMME-KNN	0.5279	0.6847	0.7152	0.9121	0.9742	0.5677	0.9745	0.8577	0.9695	0.9466	0.8164	5.9
SAMME-DT	0.5368	0.8333	0.7168	0.7595	0.8742	0.7668	0.9105	0.7157	0.8978	0.8678	0.7903	7.3
SAMME-MLP	0.5520	0.7637	0.5254	0.6952	0.9447	0.6126	0.9543	0.7223	0.9784	0.7979	0.7087	8.0

and the same train/test splits are considered. 25% of data samples in the training sets are randomly picked out to form the validation sets. These validation sets are used for identifying the best parameter settings for these ensemble classifiers aiming at maximising their performances. During the experiments, the level of granularity, G for FWAdaBoost-SOFIES, AdaBoost.M2-SOFIES and SAMME-SOFIES varies from 6, 8, 10, 12 and 14. The same five different parameter settings used in the previous example are considered for SVM, KNN, DT and MLP. The number of components within each ensemble classifier varies from 5 to 40. The obtained $BAcc$ of FWAdaBoost-SOFIES and the 10 alternative ensemble classifiers on the nine benchmark problems are tabulated in Table IV, with the best results in bold.

In addition, to evaluate the effectiveness of FWAdaBoost-SOFIES on high-dimensional problems, two visual datasets, namely, MN and FMN are used for performance comparison. To facilitate computation, 10000 training images (1000 images per class) are randomly selected to build the training sets, and 2000 training images (200 images per class) are used for validation. Classification performances on the testing sets by FWAdaBoost-SOFIES and its competitors with the optimised parameter settings are also reported in Table IV in terms of $BAcc$. The performances of the nine ensemble classifiers on each benchmark problem are ranked from best to worst, and the average ranks are given in the same table.

Similarly, pairwise Wilcoxon signed rank tests [57] are conducted to examine the statistical significance of the better performance achieved by FWAdaBoost-SOFIES over the 10 competitors on the 11 benchmark problems. The outcomes of the pairwise tests are tabulated in Supplementary Table S9. As illustrative examples, IF-THEN rules learned from the two visual datasets by ensemble components of FWAdaBoost-SOFIES are presented in Supplementary Tables S10 and S11.

Table IV shows that FWAdaBoost-SOFIES outperforms all 10 alternative ensemble classifiers on four benchmark problems, including the two high-dimensional visual ones, and its overall performance is ranked at the top. In addition, 77.3% of the p-values returned by the pairwise Wilcoxon tests as given by Supplementary Table 9 are below the level of significance specified by $\alpha = 0.05$. This example further demonstrates the superiority of FWAdaBoost-SOFIES. Visual examples given by Supplementary Tables S10 and S11 also demonstrate the high transparency of FWAdaBoost-SOFIES as the learned knowledge from data is kept in its knowledge base in the form of human-understandable, intuitive prototypes.

TABLE V
PERFORMANCE DEMONSTRATION OF FWADABOOST WITH OTHER TYPES OF BASE CLASSIFIERS

Algorithm	AB	CA	GP	IS	MF	OR	TE	Overall Acc
FWAdaBoost-ALMMo0	0.5379	0.8820	0.7658	0.8022	0.9469	0.9791	0.9900	0.8434
ALMMo0	0.4854	0.8447	0.6979	0.7689	0.9356	0.7918	0.9857	0.7871
FWAdaBoost-eClass0	0.4923	0.7662	0.4802	0.7486	0.7996	0.8919	0.8034	0.7117
eClass0	0.4587	0.7035	0.2687	0.7134	0.7918	0.8731	0.7005	0.6442

At last, to demonstrate the effectiveness of FWAdaBoost to alternative zero-order fuzzy inference systems, FWAdaBoost-ALMMo0 and FWAdaBoost-eClass0 ensemble classifiers are constructed by using ALMMo0 and eClass0 classifiers as the ensemble components, respectively, following the same ensemble framework as FWAdaBoost-SOFIES. The performances of the two ensemble classifiers are evaluated on AB, CA, GP, IS, MF OR and TE datasets, under the same experimental protocols and reported in Table V .The overall accuracy rates are given in the same table as well. One can see from Table V that FWAdaBoost effectively boosts the classification performances of ALMMo0 and eClass0, showing the applicability of FWAdaBoost to other types of zero-order fuzzy inference systems with similar operating mechanisms.

E. Discussions

Based on the numerical examples presented in this section, one could conclude that the proposed FWAdaBoost-SOFIES is a powerful ensemble approach for classification by offering users both great prediction precision and high model transparency. In particular, the ablation analysis presented in subsection V.C, namely, Fig.4 and Supplementary Table S7 demonstrate the effectiveness and validity of the new modifications introduced to the multi-class AdaBoost algorithm, resulting in the proposed FWAdaBoost. The performance comparison presented in Tables I and II justifies the efficacy of FWAdaBoost as a boosting algorithm designed specifically for SOFIS. Tables III and IV demonstrates the superior performance of FWAdaBoost-SOFIES over the state-of-the-art alternatives. Table V further shows that FWAdaBoost can be used for boosting the performances of other zero-order fuzzy inference systems such as ALMMo0 and eClass0. Nevertheless, this study has several limitations and needs to be further investigated as future work.

Firstly, numerical experiments in this paper are conducted in noise-free environments. Although it is demonstrated through numerical examples that FWAdaBoost is able to create a stronger ensemble fuzzy classifier with higher prediction precision than alternative boosting algorithms. Further experimental investigations are needed to evaluate the robustness of FWAdaBoost and examine its performance in noisy environments.

Secondly, zero-order FISs are usually linear classifiers. It is well known that the performance of a linear classifier is often very limited when applied to nonlinear problems where data samples are not linearly separable. Despite that this limitation has been partially lifted through ensemble learning, it would be worth introducing some modifications, for example, the kernel tricks to SOIFS such that the performance of FWAdaBoost-SOFIES on nonlinear classification problems can be further improved.

Last but not least, FWAdaBoost-SOFIES in this paper uses cosine dissimilarity as the default distance measure only. It is well known that distance measures play a significant role in classification. Different types of distance measures can lead to different classification results. Although cosine dissimilarity is widely used for high dimensional data classification, it is not a full distance metric and may hide important information. Therefore, it would be interesting to see how FWAdaBoost-SOFIES performs with other types of distance measures, e.g., Euclidean distance, Mahalanobis distance.

VI. CONCLUSION

This paper investigates the potential of AdaBoost to incorporate fuzzy inference systems as base classifiers and proposes FWAdaBoost-SOFIES for multi-class classification. With the modified sample weight updating and output generation schemes, FWAdaBoost is able to boost the classification performance of SOFIS to a greater extent than alternative boosting algorithms. Numerical examples demonstrated the superior performance of FWAdaBoost-SOFIES on a wide range of benchmark problems.

There are several considerations for future work. Firstly, SOFIS can learn from static data and streaming data, but FWAdaBoost-SOFIES is limited to offline application scenarios at this moment. It would be very useful to develop an online version for FWAdaBoost so that the proposed ensemble classifier can be used for time-critical applications. Secondly, introducing the kernel tricks to SOFIS and FWAdaBoost-SOFIES will be an interesting direction for future work. Such modification can improve the capability of SOFIS and FWAdaBoost-SOFIES to handle nonlinear problems. In addition, as aforementioned, the robustness of FWAdaBoost-SOFIES to noisy data needs to be investigated. Finally, the impacts of different distance measures on classification performance and robustness of FWAdaBoost-SOFIES also need to be evaluated.

REFERENCES

[1] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–44, 2006.
[2] L. Rokach, "Ensemble-based classifiers," *Artif. Intell. Rev.*, vol. 33, no. 1–2, pp. 1–39, 2010.
[3] L. Breiman, "Random forests," *Mach. Learn. Proc.*, vol. 45, no. 1, pp. 5–32, 2001.
[4] C. A. Lupascu, D. Tegolo, and E. Trucco, "FABC: retinal vessel segmentation using AdaBoost," *IEEE Trans. Inf. Technol. Biomed.*, vol. 14, no. 5, pp. 1267–1274, 2010.

[5] Z. Mian and W. Hong, "Face verification using gabor wavelets and AdaBoost," in *International Conference on Pattern Recognition*, 2006, pp. 404–407.

[6] H. Bin Shen and K. C. Chou, "Ensemble classifier for protein fold pattern recognition," *Bioinformatics*, vol. 22, no. 14, pp. 1717–1722, 2006.

[7] Y. Sun, K. Tang, L. L. Minku, S. Wang, and X. Yao, "Online ensemble learning of data streams with gradually evolved classes," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 6, pp. 1532–1545, 2016.

[8] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 421, pp. 123–140, 1996.

[9] S. Segui, L. Igual, and J. Vitria, "Weighted bagging for graph based one-class classifiers," in *International Workshop on Multiple Classifier Systems*, 2010, pp. 1–10.

[10] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[11] J. H. Friedman, "Stochastic gradient boosting," *Comput. Stat. Data Anal.*, vol. 38, no. 4, pp. 367–378, 2002.

[12] H. J. Xing and W. T. Liu, "Robust AdaBoost based ensemble of one-class support vector machines," *Inf. Fusion*, vol. 55, pp. 45–58, 2020.

[13] D. W. Opitz and R. F. MacLin, "An empirical evaluation of bagging and boosting for artificial neural networks," in *IEEE International Conference on Neural Networks*, 1997, pp. 1401–1405.

[14] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Mach. Learn.*, vol. 40, no. 2, pp. 139–157, 2000.

[15] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Mach. Learn.*, vol. 37, no. 3, pp. 297–336, 1999.

[16] C. Rudin, I. Daubechies, and R. E. Schapire, "The dynamics of AdaBoost: cyclic behavior and convergence of margins," *J. Mach. Learn. Res.*, vol. 5, pp. 1557–1595, 2004.

[17] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: a statistical view of boosting," *Ann. Stat.*, vol. 28, no. 2, pp. 337–407, 2000.

[18] S. Wu and H. Nagahashi, "Parameterized AdaBoost: introducing a parameter to speed up the training of real adaboost," *IEEE Signal Process. Lett.*, vol. 21, no. 6, pp. 687–691, 2014.

[19] A. Vezhnevets and V. Vezhnevets, "Modest AdaBoost-teaching AdaBoost to generalize better," *Graphicon*, vol. 12, no. 5, pp. 987–997, 2005.

[20] N. C. Oza, "Aveboost2: boosting for noisy data," in *International Workshop on Multiple Classifier Systems*, 2004, pp. 31–40.

[21] S. Z. Li and Z. Zhang, "FloatBoost learning and statistical face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1112–1123, 2004.

[22] J. Cao, S. Kwong, and R. Wang, "A noise-detection based AdaBoost algorithm for mislabeled data," *Pattern Recognit.*, vol. 45, no. 12, pp. 4451–4465, 2012.

[23] Z. Wang, "Robust boosting with truncated loss functions," *Electron. J. Stat.*, vol. 12, no. 1, pp. 599–650, 2018.

[24] Y. Freund, R. E. Schapire, and M. Hill, "Experiments with a new boosting algorithm," in *International Conference on Machine Learning*, 1996, pp. 148–156.

[25] R. E. Schapire and Y. Singer, "BoosTexter: a boosting-based system for text categorization," *Mach. Learn.*, vol. 39, no. 2, pp. 135–168, 2000.

[26] J. Zhu, H. Zou, S. Rosset, and T. Hastie, "Multi-class AdaBoost," *Stat. Interface*, vol. 2, no. 3, pp. 349–360, 2009.

[27] H. Schwenk and Y. Bengio, "Boosting neural networks," *Neural Comput.*, vol. 12, no. 8, pp. 1869–1887, 2000.

[28] E. Alfaro, N. Garcia, M. Gamez, and D. Elizondo, "Bankruptcy forecasting: an empirical comparison of AdaBoost and neural networks," *Decis. Support Syst.*, vol. 45, no. 1, pp. 110–122, 2008.

[29] A. Taherkhani, G. Cosma, and T. M. McGinnity, "AdaBoost-CNN: an adaptive boosting algorithm for convolutional neural networks to classify multi-class imbalanced datasets using transfer learning," *Neurocomputing*, vol. 404, pp. 351–366, 2020.

[30] X. Li, L. Wang, and E. Sung,"AdaBoost with SVM-based component classifiers," *Eng. Appl. Artif. Intell.*, vol. 21, no. 5, pp. 785–795, 2008.

[31] B. Sun, S. Chen, J. Wang, and H. Chen, "A robust multi-class AdaBoost algorithm for mislabeled noisy data," *Knowledge−Based Syst.*, vol. 102, pp. 87–102, 2016.

[32] Y. Gao and F. Gao, "Edited AdaBoost by weighted kNN," *Neurocomputing*, vol. 73, no. 16–18, pp. 3079–3088, 2010.

[33] O. Sagi and L. Rokach, "Ensemble learning: a survey," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 8, no. 4, pp. 1–18, 2018.

[34] H. Hagras, "Toward human-understandable, explainable AI," *Computer.*, vol. 51, no. 9, pp. 28–36, 2018.

[35] Z. C. Lipton, "The mythos of model interpretability," *Commun. ACM*, vol. 61, no. 10, pp. 35–43, 2018.

[36] R. Scherer, "Designing boosting ensemble of relational fuzzy systems," *Int. J. Neural Syst.*, vol. 20, no. 5, pp. 381–388, 2010.

[37] R. Scherer, "An ensemble of logical-type neuro-fuzzy systems," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 13115–13120, 2011.

[38] B. Soua, A. Borgi, and M. Tagina, "An ensemble method for fuzzy rule-based classification systems," *Knowl. Inf. Syst.*, vol. 36, no. 2, pp. 385–410, 2013.

[39] P. Angelov and X. Zhou, "Evolving fuzzy−rule based classifiers from data streams," *IEEE Trans. Fuzzy Syst.*, vol. 16, no. 6, pp. 1462–1474, 2008.

[40] J. A. Iglesias, A. Ledezma, and A. Sanchis, "Ensemble method based on individual evolving classifiers," in *IEEE Conference on Evolving and Adaptive Intelligent Systems*, 2013, pp. 56–61.

[41] M. Pratama, W. Pedrycz, and E. Lughofer, "Evolving ensemble fuzzy classifier," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 5, pp. 2552–2567, 2018.

[42] X. Gu, P. P. Angelov, C. Zhang, and P. M. Atkinson, "A massively parallel deep rule-based ensemble classifier for remote sensing scenes," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 3, pp. 345–349, 2018.

[43] A. Polyakova, L. Lipinskiy, and E. Semenkin, "Investigation of reference sample reduction methods for ensemble output with fuzzy logic-based systems," in *International Congress on Advanced Applied Informatics*, 2019, pp. 583–586.

[44] E. Alves, R. Tanscheit, and M. Vellasco, "SENFIS - selected ensemble of fuzzy inference systems," *IEEE Int. Conf. Fuzzy Syst.*, vol. 2019-June, pp. 1–6, 2019.

[45] V. Stanovov, S. Akhmedova, and Y. Kamiya, "Confidence-based voting for the design of interpretable ensembles with fuzzy systems," *Algorithms*, vol. 13, no. 4, pp. 1–15, 2020.

[46] D. Wu, C. T. Lin, J. Huang, and Z. Zeng, "On the functional equivalence of TSK fuzzy systems to neural networks, mixture of experts, CART, and stacking ensemble regression," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 10, pp. 2570–2580, 2020.

[47] X. Gu, P. Angelov, and H. J. Rong, "Local optimality of self-organising neuro-fuzzy inference systems," *Inf. Sci. (Ny).*, vol. 503, 2019.

[48] X. Gu and P. P. Angelov, "Self-organising fuzzy logic classifier," *Inf. Sci. (Ny).*, vol. 447, 2018.

[49] P. P. Angelov, X. Gu, and J. Principe, "A generalized methodology for data analysis," *IEEE Trans. Cybern.*, vol. 48, no. 10, pp. 2981–2993, 2018.

[50] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial tessellations*: *concepts and applications of Voronoi diagrams*, 2nd ed. Chichester, England: John Wiley&Sons., 1999.

[51] W. Du et al., "A new fuzzy logic classifier based on multiscale permutation entropy and its application in bearing fault diagnosis," *Entropy*, vol. 22, no. 1, p. 27, 2020.

[52] H. Zhou, H. Zhao, and Y. Zhang, "Nonlinear system modeling using self-organizing fuzzy neural networks for industrial applications," *Appl. Intell.*, vol. 50, no. 5, pp. 1657–1672, 2020.

[53] X. Gu, P. Angelov, and Z. Zhao, "Self-organizing fuzzy inference ensemble system for big streaming data classification," *Knowledge-Based Syst.*, vol. 218, p. 106870, 2021.

[54] P. Angelov and X. Gu, "Autonomous learning multi-model classifier of 0-order (ALMMo-0)," in *IEEE International Conference on Evolving and Autonomous Intelligent Systems*, 2017, pp. 1–7.

[55] P. Angelov and R. Yager, "A new type of simplified fuzzy rule-based system," *Int. J. Gen. Syst.*, vol. 41, no. 2, pp. 163–185, 2012.

[56] K. Brodersen, C. Ong, K. Stephan, and J. Buhmann, "The balanced accuracy and its posterior distribution," in *International Conference on Pattern Recognition*, 2010, pp. 3121–3124.

[57] F. Wilcoxon, "Individual comparisons of grouped data by ranking methods," *J. Econ. Entomol.*, vol. 39, no. 6, pp. 269–270, 1946.

[58] R. N. Patro, S. Subudhi, P. K. Biswal, and F. Dell'Acqua, "Dictionary-based classifiers for exploiting feature sequence information and their application to hyperspectral remotely sensed data," *Int. J. Remote Sens.*, vol. 40, no. 13, pp. 4996–5024, 2019.

[59] G. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, vol. 42, no. 2, pp. 513–529, 2012.